1. A boot method for an In-Circuit Emulation system having a microcontroller operating in lock-step synchronization with a virtual microcontroller, comprising:

in the microcontroller, executing a set of boot code;

in the virtual microcontroller, executing a set of timing code timed to take the same number of clock cycles as the microcontroller uses to execute the boot code; and

simultaneously halting both the microcontroller and the virtual microcontroller.

2. The method according to claim 1, further comprising copying register contents from the microcontroller to corresponding registers in the virtual microcontroller.

3. The method according to claim 1, further comprising copying memory contents from the microcontroller to corresponding memory in the virtual microcontroller.

4. The method according to claim 1, wherein after the executing of the boot code, the microcontroller branches to assembly instruction line 0; and wherein after executing the timing code, the virtual microcontroller branches to assembly instruction line 0.

5. The method according to claim 1, wherein prior to the executing of the boot code, and prior to executing the timing code, a break is set at assembly instruction line 0.

6. The method according to claim 1, wherein the boot code is stored within the microcontroller and hidden from the virtual microcontroller.

7. The method according to claim 1, further comprising:

prior to the executing of the boot code, and prior to executing the timing code, setting a break at assembly instruction line 0; and

wherein after the executing of the boot code, the microcontroller branches to assembly instruction line 0; and wherein after executing the timing code, the virtual microcontroller branches to assembly instruction line 0.

8. The method according to claim 1, further comprising:

prior to the executing of the boot code, and prior to executing the timing code, setting a break at assembly instruction line 0;

wherein after the executing of the boot code, the microcontroller branches to assembly instruction line 0; and wherein after executing the timing code, the virtual microcontroller branches to assembly instruction line 0;

copying register contents from the microcontroller to corresponding registers in the virtual microcontroller;

copying memory contents from the microcontroller to corresponding memory in the virtual microcontroller;

wherein after the executing of the boot code, the microcontroller branches to assembly instruction line 0; and

wherein after executing the timing code, the virtual microcontroller branches to assembly instruction line zero.

9. The method according to claim 8, further comprising removing the break at assembly line zero after copying the register contents and copying the memory contents.

10. A boot method for an In-Circuit Emulation system having a microcontroller operating in lock-step synchronization with a virtual microcontroller, comprising:

resetting the microcontroller and the virtual microcontroller to a halt state;

setting a break at assembly instruction line zero;

in the microcontroller, executing a set of boot code;

in the virtual microcontroller, simultaneously executing a set of timing code timed to take the same number of clock cycles as the microcontroller uses to execute the boot code;

simultaneously halting both the microcontroller and the virtual microcontroller by branching to assembly instruction line zero;

copying register contents from the microcontroller to corresponding registers in the virtual microcontroller;

copying memory contents from the microcontroller to corresponding memory in the virtual microcontroller; and

removing the break at assembly line zero after copying the register contents and copying the memory contents.

11. The method according to claim 10, wherein the boot code is stored within the microcontroller and hidden from the virtual microcontroller.

12. A boot method for an In-Circuit Emulation system having a device under test operating in lock-step synchronization with a virtual processor, comprising:

in the device under test, executing a set of boot code;

in the virtual processor, executing a set of timing code timed to take the same number of clock cycles as the device under test uses to execute the boot code; and

simultaneously halting both the device under test and the virtual processor.

13. The method according to claim 12, further comprising copying register contents from the device under test to corresponding registers in the virtual processor.

14. The method according to claim 12, further comprising copying memory contents from the device under test to corresponding memory in the virtual processor.

15. The method according to claim 12, wherein after the executing of the boot code, the device under test branches to assembly instruction line 0; and wherein after executing the timing code, the virtual processor branches to assembly instruction line 0.

16. The method according to claim 12, wherein prior to the executing of the boot code, and prior to executing the timing code, a break is set at assembly instruction line 0.

17. The method according to claim 12, wherein the boot code is stored within the microcontroller and hidden from the virtual microcontroller.

18. The method according to claim 12, further comprising:

prior to the executing of the boot code, and prior to executing the timing code, setting a break at assembly instruction line 0; and

wherein after the executing of the boot code, the device under test branches to assembly instruction line 0; and wherein after executing the timing code, the virtual processor branches to assembly instruction line 0.

19. The method according to claim 12, further comprising:

prior to the executing of the boot code, and prior to executing the timing code, setting a break at assembly instruction line 0;

wherein after the executing of the boot code, the device under test branches to assembly instruction line 0; and wherein after executing the timing code, the virtual processor branches to assembly instruction line 0;

copying register contents from the microcontroller to corresponding registers in the virtual processor;

copying memory contents from the device under test to corresponding memory in the virtual microcontroller;

wherein after the executing of the boot code, the microcontroller branches to assembly instruction line 0; and

wherein after executing the timing code, the virtual processor branches to assembly instruction line zero.

20. The method according to claim 19, further comprising removing the break at assembly line zero after copying the register contents and copying the memory contents.

21. The method according to claim 12, wherein the virtual processor is implemented in a field programmable gate array.